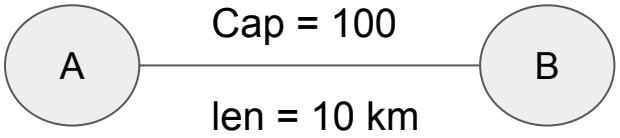
Assignment 4

Graph

- Vertices
  - Cities - No capacity
  - Shelters - Limited capacity
- Edges
  - Length
  - Capacity
- Properties
  - Disconnected graphs
  - Multiple edges between 2 vertices
  - Undirected

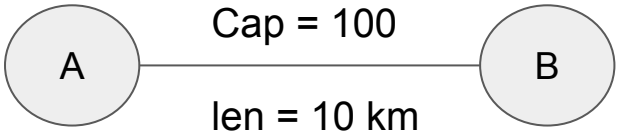
Movement



There are 250 people in City A.

- The first group of 100 moves from A to B.
- The second group of 100 follows, moving from A to B.
- Finally, the last 50 relocate from A to B.

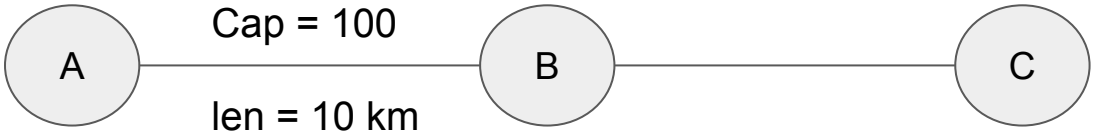
Movement



People move at constant pace of 5km/hr

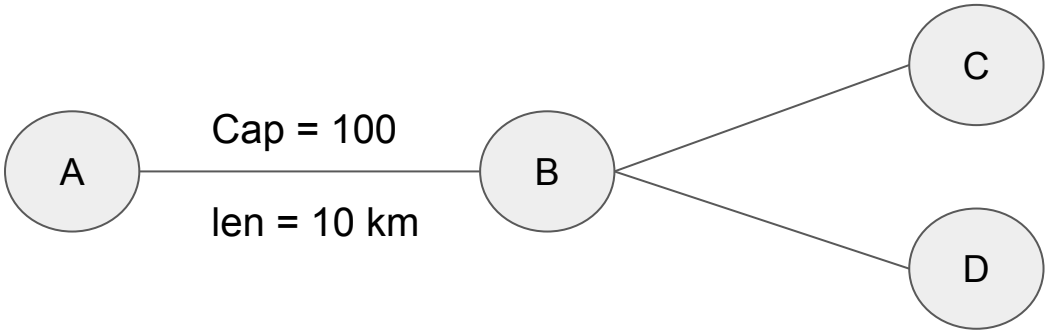
- 2 hrs per group
- 6 hrs for 3 groups

Movement



People can move from B to C only after complete movement from A to B is complete

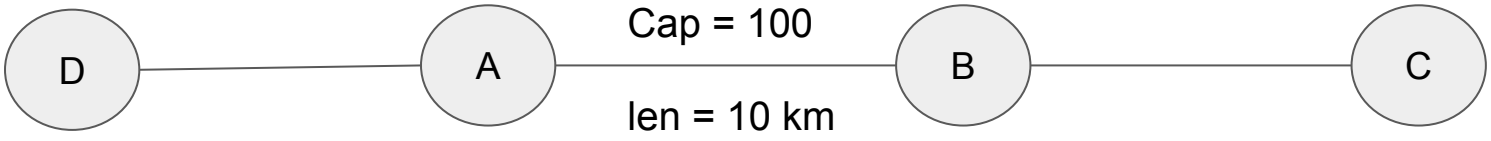
Movement



Cannot split people along different paths
Eg 100 from B to C

and 100 from B to D

Movement



People already moving from A to B

- People from other city who have arrived at A or B will have to wait.
- In case people from 2 cities come simultaneously and want to use same path then priority will be given to more population and if population is same then lower city number will be given priority.

Drops

- People can be dropped in any city/shelter
- Only people who are present in shelter will be counted as “saved” people.
- If shelter has a capacity of 100 and you drop 150 people
  - Additional 50 people won’t be counted as saved
  - Also a penalty will be applied - same number of people from the shelter will be considered as dead.
  - So only 150 - 50 - 50 = 50 people will considered as “saved”
  - Number of people saved = max(0, shelter capacity - additional people)

Expected output

- If there are n populated cities then n paths.
- Drops along n paths
  - Drop cities/shelters must appear in same order as they appear in path
  - You have to store city number, prime-age people dropped and old people.

$$\text{score} = 0.6 \times \left(\frac{P - P_{\min} + \alpha}{P_{\max} - P_{\min} + \alpha} \right) + 0.4 \times \left(\frac{T_{\max} - T + \beta}{T_{\max} - T_{\min} + \beta} \right)$$

$$S' = \frac{S}{S_{\max}}$$

Advice

- Implement brute force first - then try variations of Shortest path/MST
- Try to save as many people as possible
- Notice that your solution is valid as long as you are dropping all the people and you have saved at least 1 person. Try to print valid paths in worst case
- Use parallelism as much as possible
- Use thrust wherever possible
- Read up about mallocs, free on GPU and remain careful about memory management
- Read the problem statement carefully to ensure that you are correctly handling all the constraints.